THE LIGHTNING MAN ALIVE: 3D GAME

SAKSHI GUPTA
Computer Science and Engineering
Moradabad Institute of Technology
(of AKTU)
Moradabad, Uttar Pradesh, India
sakshiigupta10@gmail.com

MANPREET SINGH
Computer Science and Engineering
Moradabad Institute of Technology
(of AKTU)
Moradabad, Uttar Pradesh, India
mpschahal16@gmail.com

PRANJAL GUPTA
Computer Science and Engineering
Moradabad Institute of Technology
(of AKTU)
Moradabad, Uttar Pradesh, India
pranjal2001gupta@gmail.com

PRAJJWAL BHARDWAJ
Computer Science and Engineering
Moradabad Institute of Technology
(of AKTU)
Moradabad, Uttar Pradesh, India
vermaprjwl@gmail.com

ABSTRACT— *The paper is on the development of the Computer-based 3-D game. In this paper, we present the design and implementation of a 3-D game which we will develop, called THE LIGHTNING MAN ALIVE. In this game, there are two characters in this 3-D game i.e. male character named "Flash" and female named "Nora", both are infinite runners having lightning power with them which makes them to run faster, faster and more faster. The game is available in two different modes that are Light mode and Dark mode. The game is divided into levels and difficulty of the levels get increased as much as you qualify levels or we can say level by level difficulty level will increase. In this game, the task is to collect coins as more as you can in given time limit by crossing hurdles in form of cars and trucks. As you cover more distance, your speed also get increased.*
The tool used for developing this game is Unity 3D game engine. Systems will be able to successfully run the game. We hope that sharing our experiences will assist others who wish to either use our computer-based game or develop their own. Our research seeks to adding values and enjoyment to users of the systems.

Key Words: Game Engines; Animation; Scripting; Mechanics.

I. INTRODUCTION

"The Lightning Man Alive" is an action based 3D infinite running game where the player takes the role of the main character to cross all hurdles and complete the task. The game is inspired from a famous series "The Flash", which tells the story of the last man in the world where entire humanity has returned into mindless human killers. The player experiences the game world through the eyes of a main character. This makes one feel one is a part of the game. The player controls the main character, who is one of the healthy survivors. The abilities of our character as a player are walking and running in all four directions. The player starts the game with a weapon of lightening. The effect of damage is collision with hurdles. In this game, the task is to collect coins as more as you can in given time limit by crossing hurdles in form of cars and trucks. As you cover more distance, your speed also get increased.

Casual gamers tend to enjoy simple, yet dynamic games that are easy to understand, frequently reward the player, are short in duration (as opposed to games where characters must be developed and nurtured), and have high re-playability by not becoming boring or repetitive. Our game achieves all these goals by adding new game play features and tiles as the level progresses.

II. LITERATURE SURVEY

- Study of Unity 3D game engine for use of its physics and texture engines.
- Use of Blender and Unity 3d-Max for model and background design.
- Now-a-days many big companies have developed huge and interesting computer-based games that have set the bar high.
- In this independent game with our limited resources we will attempt to recreate the same spark as games developed by these big games.

III. GAME ENGINES

A game engine is a software framework designed for the creation and development of video games.
The core functionality typically provided by a game engine includes:

- Rendering engine ("renderer") for 2D or 3D graphics
- Physics engine or collision detection
- Animation
- Scripting
- Modeling
- Texturing

a. Unity-Game Engine

Unity is a game development ecosystem: a powerful rendering engine fully integrated with a complete set of intuitive tools and rapid workflows to create interactive 3D and 2D content [1]. Unity provides all the tools a game developer need to develop a 2D or 3D game on multiple platforms with minimum cost and time which makes it ideal for independent developers like us.

Unity is a powerful engine with a variety of tools that can be utilized to meet your specific needs. The editor is intuitive and customizable allowing you a greater freedom in your workflow. This section is the key to getting started with Unity. It will follow through the important steps for creating a game in Unity; starting with the asset workflows, then how to build up your scenes and finally to publishing your build.

b. Blender-Game Engine

Blender is the free and open source 3D creation suite. It supports the entirety of the 3D pipeline—modeling, rigging, animation, simulation, rendering, compositing and motion tracking, even video editing and game creation. Advanced users employ Blender's API for Python scripting to customize the application and write specialized tools; often these are included in Blender's future releases. Blender is well suited to individuals and small studios who benefit from its unified pipeline and responsive development process.

Blender is cross-platform and runs equally well on Linux, Windows, and Macintosh computers. Its interface uses OpenGL to provide a consistent experience. To confirm specific compatibility, the list of supported platforms indicates those regularly tested by the development team. As a community-driven project under the GNU General Public License (GPL), the public is empowered to make small and large changes to the code base, which leads to new features, responsive bug fixes, and better usability. Blender has no price tag, but you can invest, participate, and help to advance a powerful collaborative tool: Blender is your own 3D software [6].

c. 3d-Max-Game Engine

3Ds Max is a powerful 3D modeling, animation, effects, and rendering solution that has been used in everything from video games to feature films. 3Ds Max has a wealth of features that can tackle almost any sort of project and generate incredibly realistic or highly stylized images. 3Ds Max's large feature set may seem daunting to the newcomer, but the software has a consistent and easy-to-use interface, and this chapter introduces that interface. Autodesk 3ds Max, formerly 3D Studio Max, is a professional 3D computer graphics program for making 3D animations, models, games and images. It is developed and produced by Autodesk Media and Entertainment. It has modeling capabilities, a flexible plugin architecture and can be used on the Microsoft Windows platform. It is frequently used by video game developers, many TV commercial studios and architectural visualization studios. It is also used for movie effects and movie pre-visualization. In addition to its modeling and animation tools, the latest version of 3ds Max also features shades, dynamic simulation, particle systems, radiosity, normal map creation and rendering, global illumination, a customizable user interface, and its own scripting language[2].

IV. OVERVIEW OF GAME

The features of our developed game are as follows:

- **Start Window of the Game:** The player experiences the game world through the eyes of a main character. This makes one feel one is a part of the game. The player controls the main character, who is one of the healthy survivors. The abilities of our character as a player are walking and running in all four directions. The player starts the game with a weapon of lightening. The effect of damage is collision with hurdles.

- **Choose Characters (Male or Female):** You have to choose character by clicking on button "CHOOSE CHARACTER". There is a male character and a female character.

- **Environment Type of the Game (Dark or Light):** You can play this 3-D game in two modes:
  - LIGHT Mode
  - DARK Mode

- **Game Sound and UI Sound:** You can adjust Game sound and UI sound accordingly by green bars available on Settings panel.

- **Play the Game:** You can play the game by clicking on PLAY button which is available on Start window. The character which you have chosen starts running on the road and execute given task. Coins collected, Current Score and Level has been also shown to the player while playing.

- **Resume/Exit/Back Main Menu:** To Resume the Game:
  - Click on RESUME button.
  - To Continue the Game: Click on START AGAIN button.
  - To go back to Main Menu: Click on BACK MAIN MENU button.
  - To Exit the Game: Click on EXIT button.
  - You can easily choose the button by clicking on it from the window.

- **Score Card of Current Level in Game:** The score of the player will be shown when the player collide with a hurdle. There will be two scores shown to the player:
  i. Current Score
  ii. High Score

- **Environment of the Game:** Environment of the game includes:
  - Buildings
  - Tress
  - Benches
  - Roads
  - Grass
  - Cars and Trucks

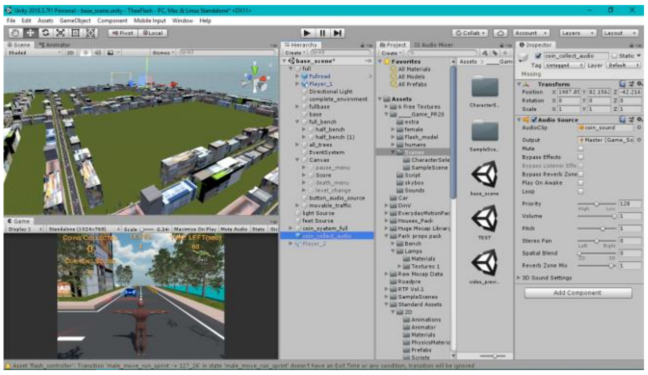
Fig 4.1 Build-up Scene in Unity

V. PRODUCTION OF GAME ASSETS

3d-Max is an object oriented program. This means that each objects in the 3d scene carries instructions that tell 3d Max what you can do with it. These instructions vary with the type of objects. Because each object can respond to different set of commands, the commands are applied by first selecting the object and then selecting the command. This is known as a noun-verb interface, because first the object (the noun) is selected and then the command (the verb).
Objects used to create models include “Geometry” and “Shapes”.
Objects used to set up the scene include “Lights” and “Cameras”.
To create an object model, follow these steps:

- Name your materials, textures and geometry sensibly.
- Use normal maps for extra detail.
- Place textures in assets\object name\textures within Unity: It’s good to have your textures set up in a pipeline that makes them convenient to work with on your project, and a good size to work with for your builds. For example, some artists generate enormous PSDs, which live outside of the Assets directory, and then from those PSDs, generate the PNGs that reside in the Assets directory in the project.
- Re-path textures before you export if required.

To develop objects on game engine:

Step-1: Rig the Game Object.
Step-2: Verify and Export.
Step-3: Import the model.
Step-4. Set-up your material.
Step-5. Create an Avatar.
Step-6. Add a Character Controller and cameras.
Step-7. Add your Animation.
Step-8. Add Blend Shapes and tweak your character.
Step-9. Add environment elements, lights and special effects.

VI. UVW MAPPING, TEXTURING, ANIMATION

UVW Mapping: UVW mapping is a mathematical technique for coordinate mapping. The UVW mapping is suitable for painting an object's surface based on a solid texture. This is useful for showing a vase carved out of the marble rock. "UVW", like the standard Cartesian coordinate system, has three dimensions; the third dimension allows texture maps to wrap in complex ways onto irregular surfaces. Each point in a UVW map corresponds to a point on the surface of the object. The graphic designer or programmer generates the specific mathematical function to implement the map, so that points on the texture are assigned to (XYZ) points on the target surface. Generally speaking, the more orderly the unwrapped polygons are, the easier it is for the texture artist to paint features onto the texture [3]. Once the texture is finished, all that has to be done is to wrap the UVW map back onto the object, projecting the texture in a way that is far more flexible and advanced, preventing graphic artifacts that accompany more simplistic texture mappings such as planar projection. For this reason, UVW mapping is commonly used to texture map non-platonic solids, non-geometric primitives, and other irregularly shaped objects, such as characters and environment.

Texturing: The UV/Image Editor allows you to map textures directly to the mesh faces. The 3D View editor shows you the object being textured. If you set this editor into textured viewport shading, you will immediately see any changes made in the UV/Image and this editor, and vice versa. You can edit and load images, and even play a game in the Blender Game Engine with UV textures for characters and object, without a material, and still see them in the 3D View. This is because no real rendering is taking place; it is all just viewport shading. If you were to apply an image to UVs then render, the texture would not show up by default.

In the Texture channel panel, add a New Texture and define the texture as an image and load the image you want to use. In the Mapping section, choose UV from the Coordinates menu, and select the UV map to use. Make sure it is mapped to Color in the Influence section as well (it will be mapped to Color by default, and the UV texture is named “UVTex” by default) [1].

Animation: Animation is possibly the most difficult art form anyone can attempt. It takes years to get good at it and one is never done learning. It is also impossible to be “taught” animation. One can be shown how to make a model or do an effect but unless one practice and just go off on his own, one, will never truly know how to do animation. Much like programming and other computer related tasks, animation is “self-taught”.

Animation is making an object move or change shape over time. Objects can be animated in many ways:

- **Moving as a whole object:** Changing their position, orientation or size in time;
- **Deforming them:** Animating their vertices or control points.
- **Inherited animation:** Causing the object to move based on the movement of another object (e.g. its parent, hook, armature, etc.).

Object animation is a form of stop motion animation that involves the animated movements of any non-drawn objects such as toys, blocks, dolls, etc. which are not fully malleable, such as clay or wax, and not designed to look like a recognizable human or animal character[3]. Object animation is considered a different form of animation distinct from model animation and puppet animation, as these two forms of stop-motion animation usually use recognizable characters as their subjects.

VII. SCRIPTING OF GAME

Scripting is an essential ingredient in all games. Even the simplest game needs scripts, to respond to input from the player and arrange for events in the gameplay to happen when they should. Beyond that, scripts can be used to create graphical effects, control the physical behavior of objects or even implement a custom AI system for characters in the game.

Scripting is a skill that takes some time and effort to learn. The intention of this section is not to teach you how to write script code from scratch, but rather to explain the main concepts that apply to scripting in Unity. Although Unity uses an implementation of the standard Mono runtime for scripting, it still has its own practices and techniques for accessing the engine from scripts [5]. This section explains how objects created in the Unity editor are controlled from scripts, and details the relationship between Unity's gameplay features and the Mono runtime.

A list of elements might an object follows:

- Transform
- Renderer
- Physics
- Colliders
- Timeline and Correction Window
- Input

VIII. CONCLUSION

Concerning the managerial aspects, the member acquired experience in how to apply an iterative process for game development. Due to the fact that the group consists of four members, all have the responsibility of all the roles for continuity of the project. Therefore, all team members participated in producing each document. The responsibility of steering the group and keeping the project on track is on all members' hands. The result of the first iteration of the process is the documentation about project planning that is necessary for the successful progress. Because of lack of experience in game development, we faced some difficulties while producing the game design document. All the members of the group must have overall idea of the game concept and mechanics clearly to do implementation. At the beginning, we could not estimate which properties could be implemented within project time. Due to time shortage, we had to put 3D graphics design and creation out of project scope. The testing process of this project went mostly as we had planned; however the test plan schedule was pretty much followed. Because implementation took longer time as we expected, we had to spend less time on testing process but we maintained that too. The end result of the implementation is a game that almost fulfills all of its functional requirements.

REFERENCES

[1] Orland, Kyle "How new graphics effects can make Unity Engine games look less generic"

[2] Hejlsberg, Anders. "Future directions for C# and Visual Basic". C# lead architect. Microsoft

[3] Prudom, Laura "The Flash: Robbie Amell Cast as Firestorm". Variety Archived from the original on July 13

[4] Brodkin, Jon "How Unity3D Became a Game-Development Beast". Dice Insights.

[5] Mullen, T (2009). Mastering Blender. 1st ed. Indianapolis, Indiana: Wiley Publishing, Inc.

[6] Blenderfoundation "Blender 2.58a update log" blender.org